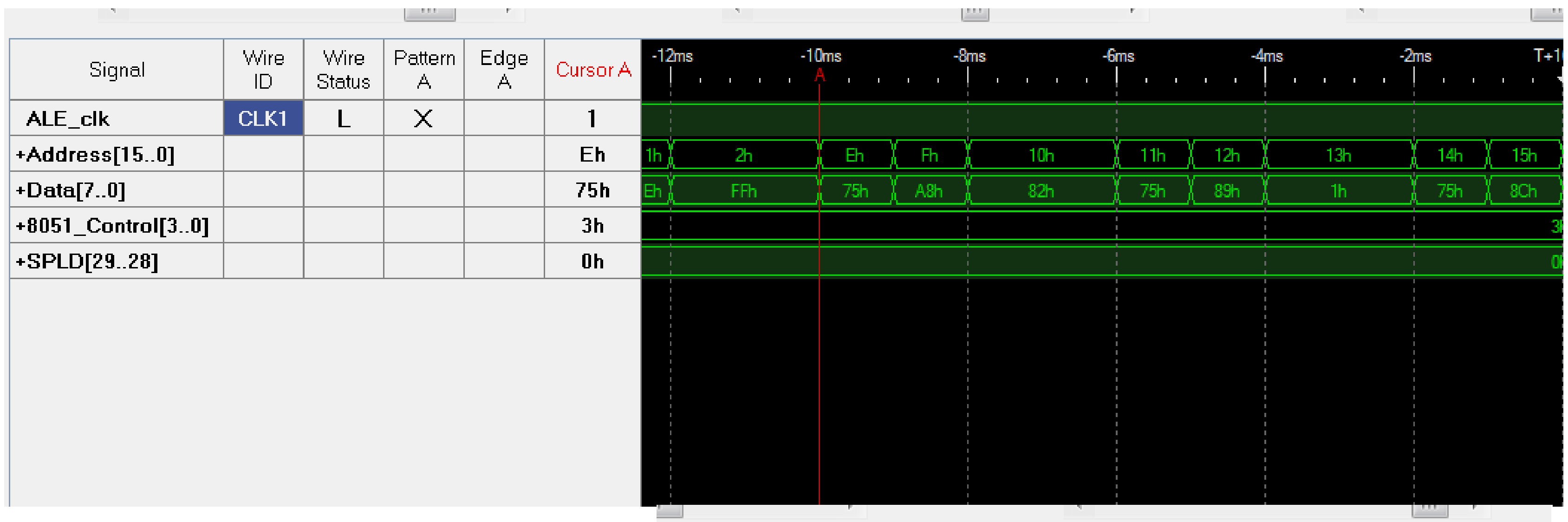
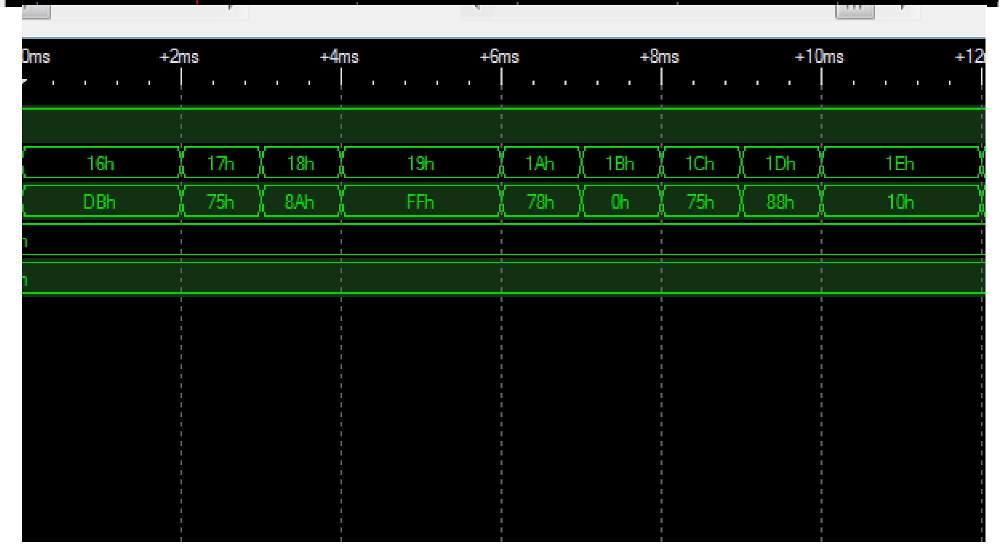


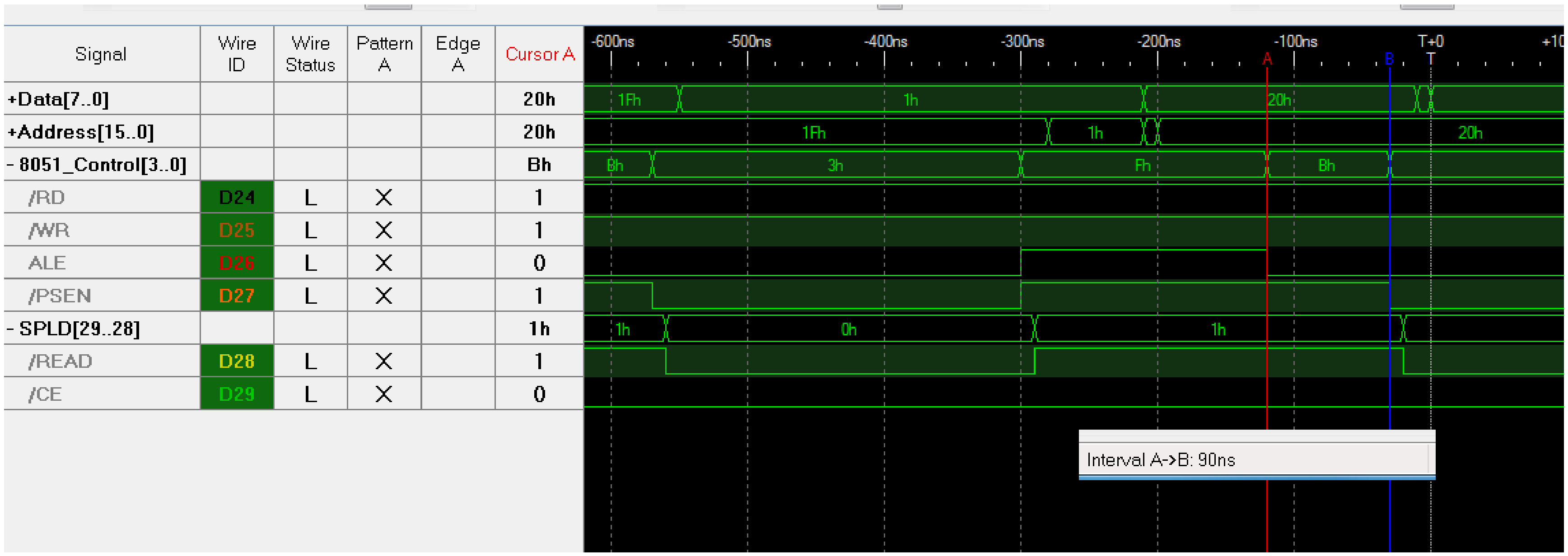
Signal	Wire ID	Wire Status	Pattern A	Edge A	Cursor A
ALE_clk	CLK1	L	X		1
+Address[15..0]					Eh
+Data[7..0]					75h
+8051_Control[3..0]					3h
+SPLD[29..28]					0h

-12ms
-10ms
A
-8ms
-6ms
-4ms
-2ms
1h 2h Eh Fh 10h 11h 12h 13h 14h 15h
Eh FFh 75h A8h 82h 75h 89h 1h 75h 8Ch



0ms
+2ms
+4ms
+6ms
+8ms
+10ms
16h 17h 18h 19h 1Ah 1Bh 1Ch 1Dh 1Eh
DBh 75h 8Ah FFh 78h 0h 75h 88h 10h

This screen capture shows the instructions being loaded out of the NVRAM. The logic analyzer was set in state mode to clock on the rising edge of the ALE and to trigger on the beginning address of the program. If you compare the address with the corresponding data value, you will see that it matches the address and opcode present at that address in the listing file.



Signal	Wire ID	Wire Status	Pattern A	Edge A	Cursor A
+Data[7..0]					20h
+Address[15..0]					20h
- 8051_Control[3..0]					Bh
/RD	D24	L	X		1
/WR	D25	L	X		1
ALE	D26	L	X		0
/PSEN	D27	L	X		1
- SPLD[29..28]					1h
/READ	D28	L	X		1
/CE	D29	L	X		0

Interval A->B: 90ns

This is a screen capture in timing mode. Cursor A is when the ALE goes low and cursor B when the PSEN signal goes low. This represents the time t_{LLPL}, which is basically the time the address value is valid after the ALE has gone low. The minimum time for this value is:

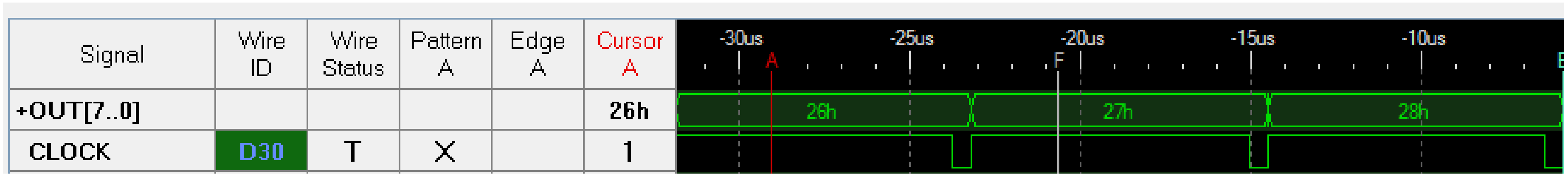
$$t_{clcl} = \frac{1}{11{,}059{,}200} = 90.4224\ ns$$

$$Minumum\ t_{LLPL} = t_{clcl} - 25 = 65.4224$$

Since the measured t_{LLPL} was 90 ns, then this means the timing requirement has been met.

Supplemental Turn In:

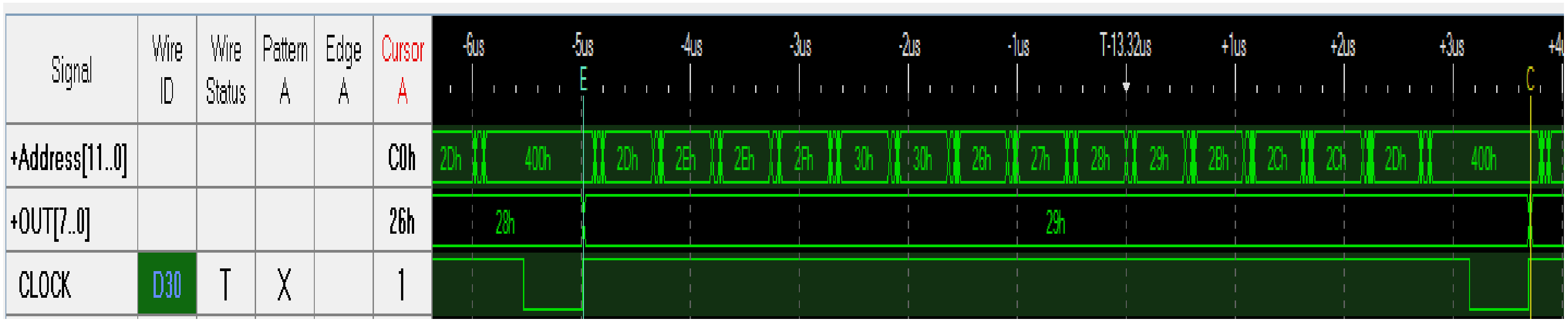
The goal for the supplemental part was to increment and latch a count value for each routine running in a program. To achieve this, I placed a counter in each routine in the program and increment every iteration of that program. In order to latch the counter value, each routine also does a write to memory. While nothing will be written to memory, the count value will be placed on the Port 0 bus allowing it to be latched by the 374’ latch. The 374’ latch is being clocked by the /WR signal. This active low signal indicates when the count value is on the port 9 bus and the latch will capture this value. In addition, the latch is clocked on the A15 address. This forces the latch to only capture counts for the address space between 0-7fffh.



Signal	Wire ID	Wire Status	Pattern A	Edge A	Cursor A
+OUT[7..0]					26h
CLOCK	D30	T	X		1

Figure 1

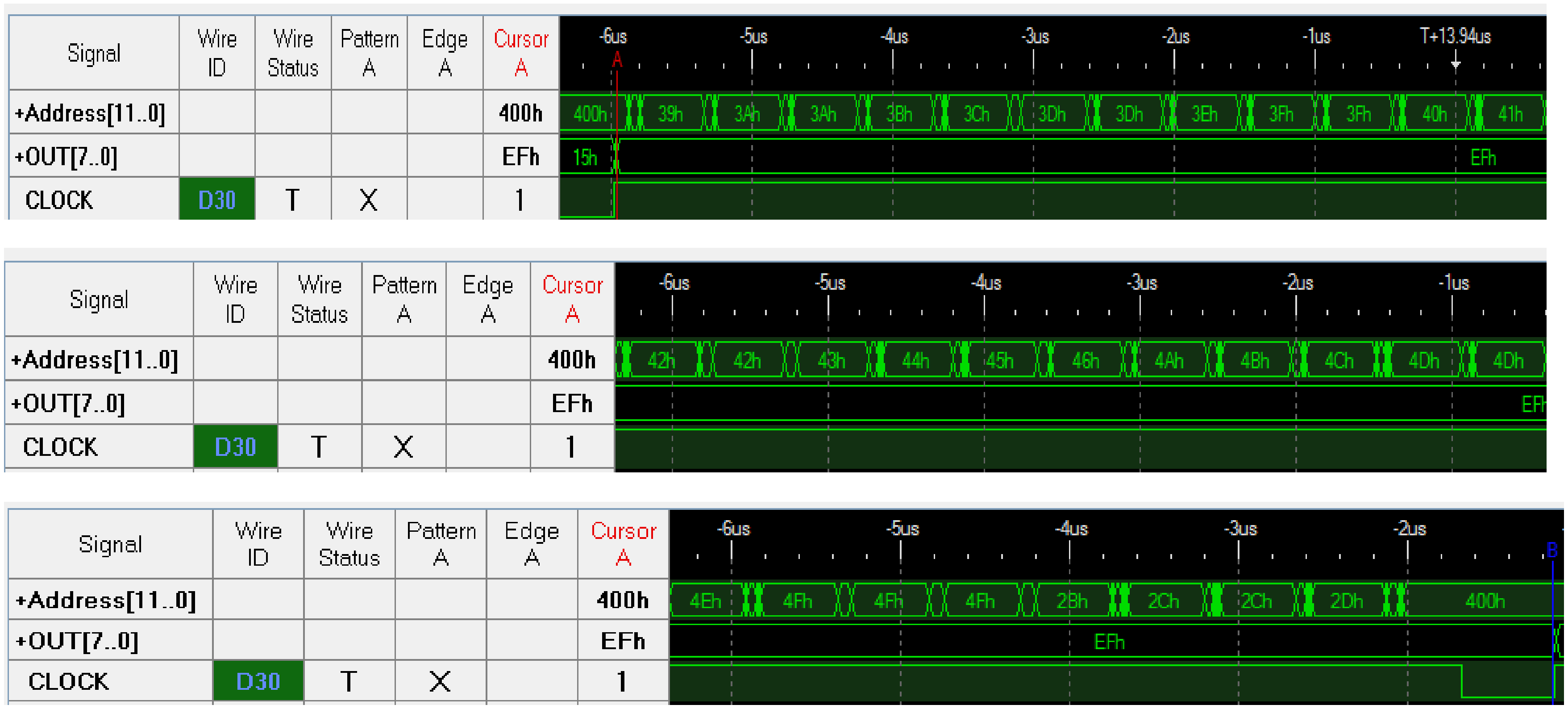
The first step in the supplemental section is to verify the functionality of the algorithm described above. In figure 1, you can see the OUT signal incrementing. OUT is the output of the 374 latch. It is currently outputting an incrementing value in an expected range. This is a slight indication that the program is working correctly.



Signal	Wire ID	Wire Status	Pattern A	Edge A	Cursor A
+Address[11..0]					C0h
+OUT[7..0]					26h
CLOCK	D30	T	X		1

Figure 2

To thoroughly verify the supplemental algorithm, I took a closer look at where the program was in its routine. During a latch, the program should only be running in a single routine. I compared the addresses in figure 2 to the addresses in my listing file. The latched value corresponds to the correct sequence of instructions. 400h represents the place in memory where a routine writes to memory and indicates when the main program puts its count value on the Port 0 bus. From addresses 2Dh to 30h is the rest of main. 26h to 400h is the beginning of main to the next write to memory. The OUT value is 29h, which is in between 0h-7fh and verifies that the 374 is latching at the correct time and the correct value. Note that some of the addresses may repeat due to the high sampling rate of the logic analyzer.



Signal	Wire ID	Wire Status	Pattern A	Edge A	Cursor A
+Address[11..0]					400h
+OUT[7..0]					EFh
CLOCK	D30	T	X		1

Signal	Wire ID	Wire Status	Pattern A	Edge A	Cursor A
+Address[11..0]					400h
+OUT[7..0]					EFh
CLOCK	D30	T	X		1

Signal	Wire ID	Wire Status	Pattern A	Edge A	Cursor A
+Address[11..0]					400h
+OUT[7..0]					EFh
CLOCK	D30	T	X		1

Figure 3

The same analysis on the main routine was done for the ISR routine. Figure 3 contains three captures of the entire instruction sequence for the ISR. As you can see, the value EFh was latched. This value is between 80h-FFh, which is the correct range for the ISR. In addition, the addresses on the address line are also in the ISR routine and running in the correct sequence. 400h indicates the write to memory and the ISR continues to the end at 4Fh. The next value is not latched until the write next to memory in the main routine, thus, the beginning of the main routing is present in the address line.